

# Model distillation and extraction

CS 685, Fall 2021

Advanced Natural Language Processing

Mohit Iyyer

College of Information and Computer Sciences

University of Massachusetts Amherst

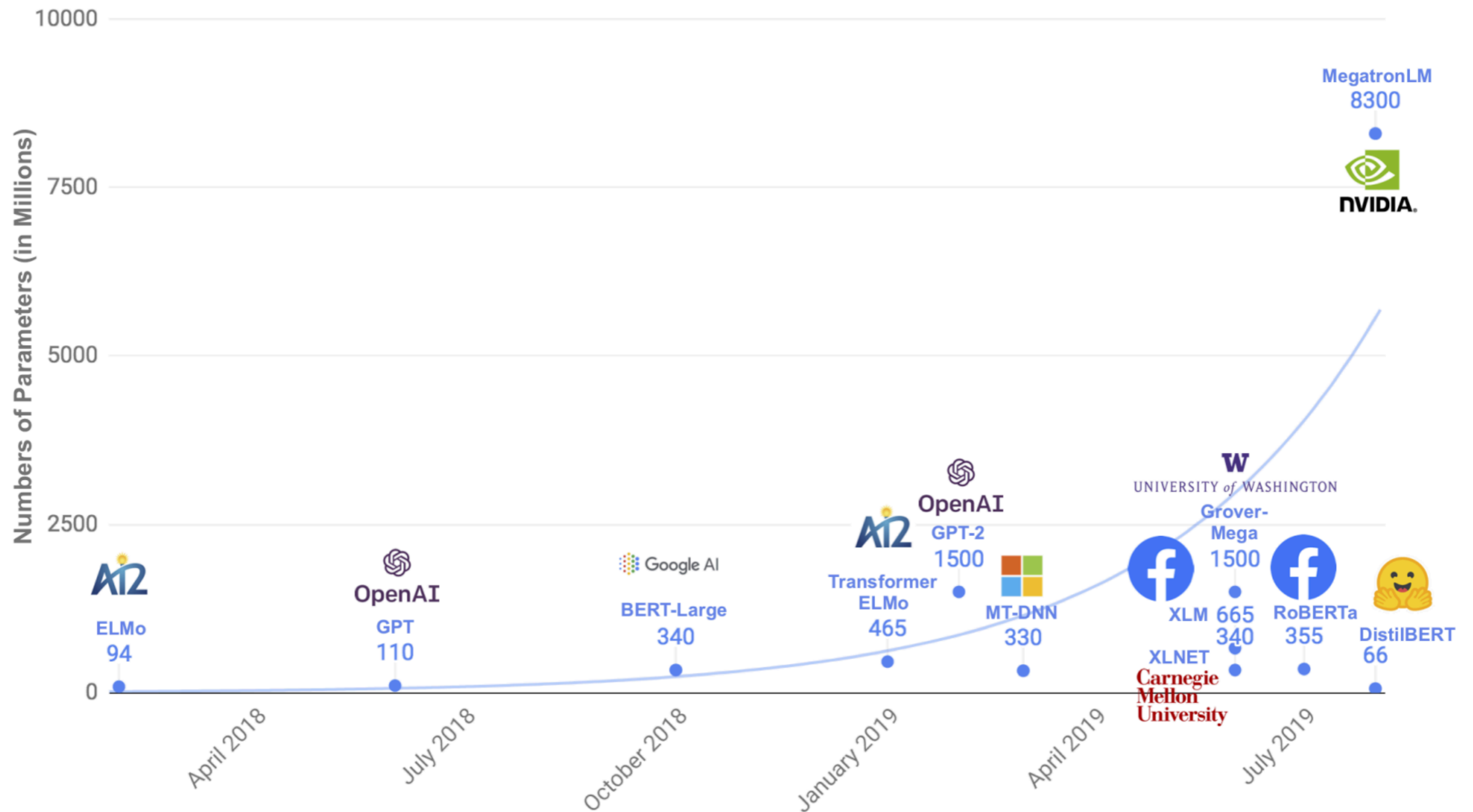
*many slides from Kalpesh Krishna*

# stuff from last time...

- Exam grades to be released tmrw, regrades on for one week
  - note that scores can go *down* as well if you submit a regrade request
- Final project reports due Dec 16, 11:59pm
  - Report PDF due on Gradescope, code via email to instructors account with README
    - Email with either **public** GitHub link or zip file is fine

# Knowledge distillation:

A small model (the **student**) is trained to mimic the predictions of a much larger pretrained model (the **teacher**)



**Figure 1: Parameter counts of several recently released pre-trained language models.**

Bob went to the <MASK>  
to get a buzz cut



barbershop: 54%  
barber: 20%  
salon: 6%  
stylist: 4%  
...

Bob went to the <MASK>  
to get a buzz cut



barbershop: 54%  
barber: 20%  
salon: 6%  
stylist: 4%  
...

soft targets

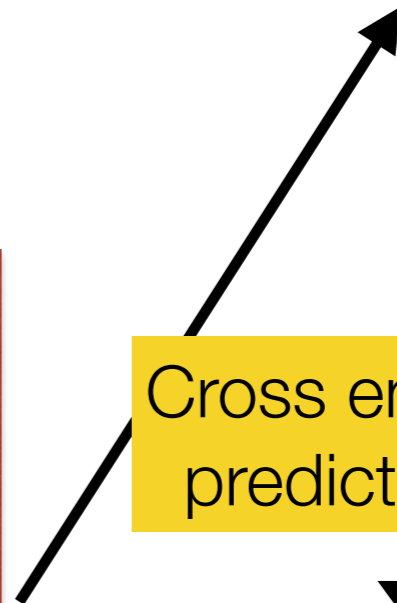
Bob went to the <MASK>  
to get a buzz cut



barbershop: 54%  
barber: 20%  
salon: 6%  
stylist: 4%  
...

soft targets  $t_i$

Bob went to the <MASK>  
to get a buzz cut



Cross entropy loss to  
predict *soft targets*

$$L_{ce} = \sum_i t_i \log(s_i)$$

# Instead of “one-hot” ground-truth, we have a full predicted distribution

- More information encoded in the target prediction than just the “correct” word
- Relative order of even low probability words (e.g., “church” vs “and” in the previous example) tells us some information
  - e.g., that the <MASK> is likely to be a noun and refer to a location, not a function word



**Table 1: DistilBERT retains 97% of BERT performance.** Comparison on the dev sets of the GLUE benchmark. ELMo results as reported by the authors. BERT and DistilBERT results are the medians of 5 runs with different seeds.

Model	Score	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST-2	STS-B	WNLI
ELMo	68.7	44.1	68.6	76.6	71.1	86.2	53.4	91.5	70.4	56.3
BERT-base	79.5	56.3	86.7	88.6	91.8	89.6	69.3	92.7	89.0	53.5
DistilBERT	77.0	51.3	82.2	87.5	89.2	88.5	59.9	91.3	86.9	56.3

# Can also distill other parts of the teacher, not just its final predictions!

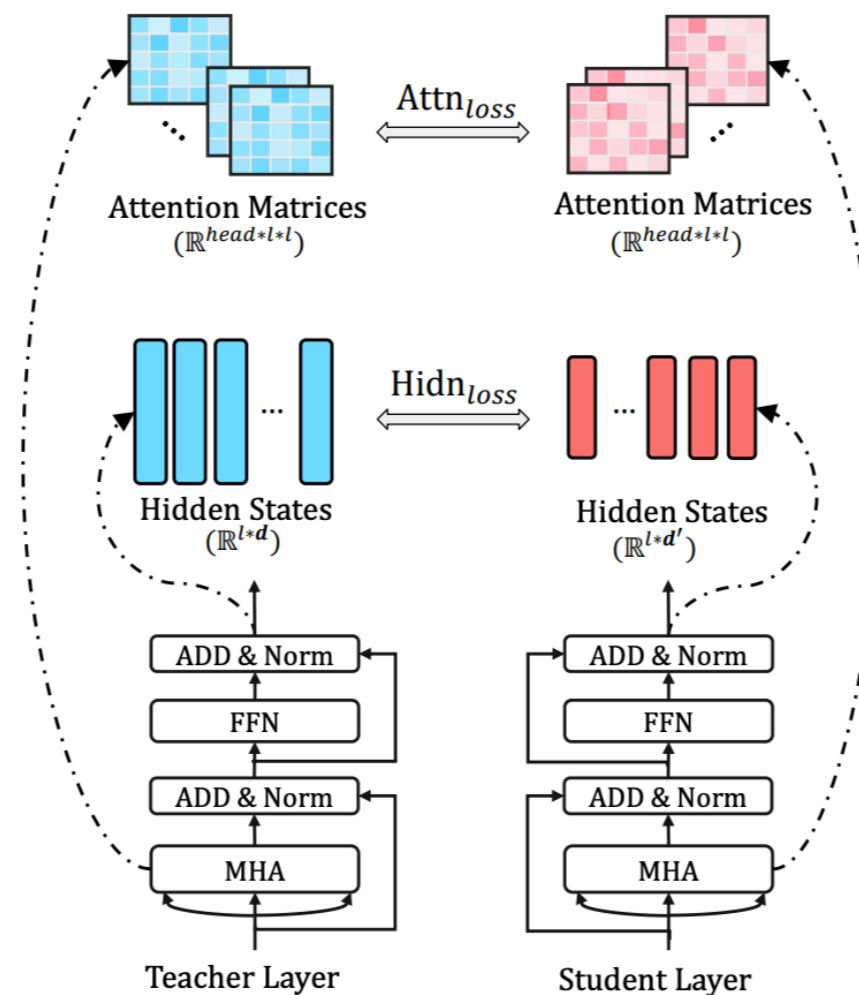
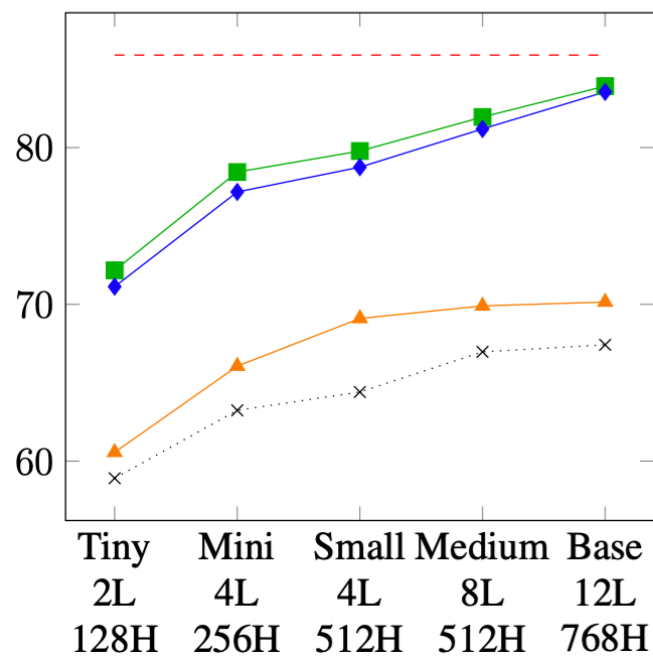


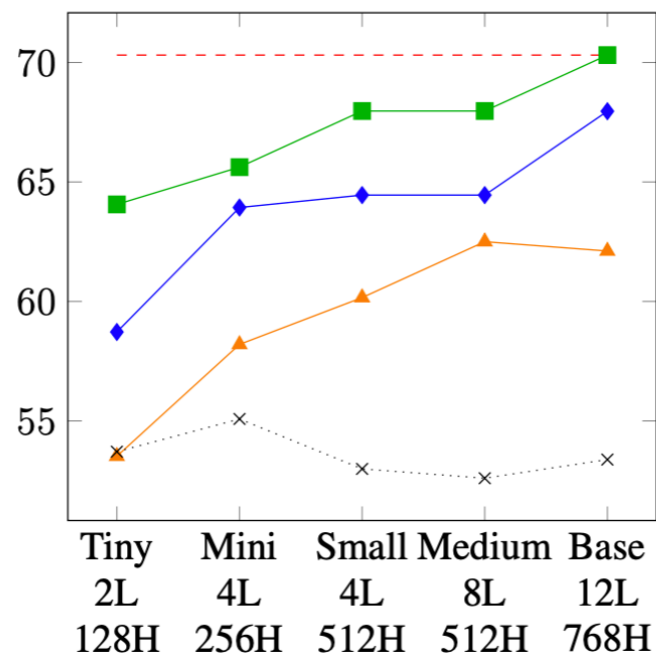
Figure 2: The details of Transformer-layer distillation consisting of  $\text{Attn}_{loss}$  (attention based distillation) and  $\text{Hidn}_{loss}$  (hidden states based distillation).

# Distillation helps significantly over just training the small model from scratch

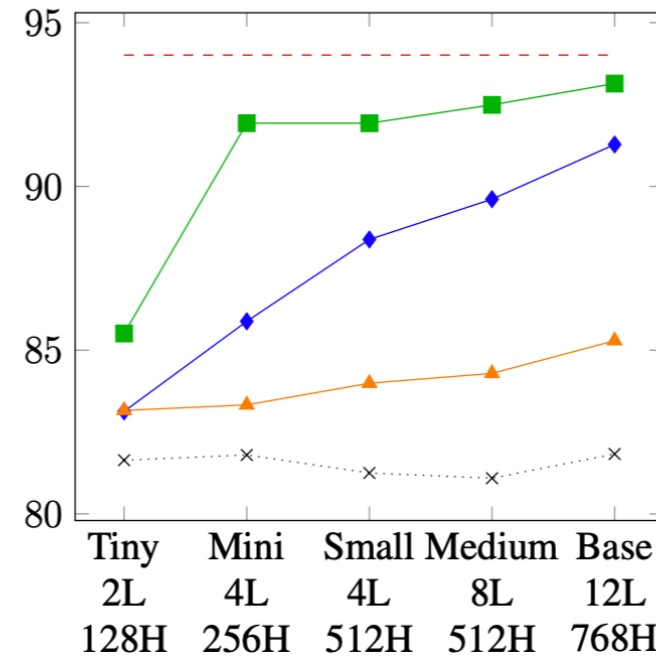
MNLI



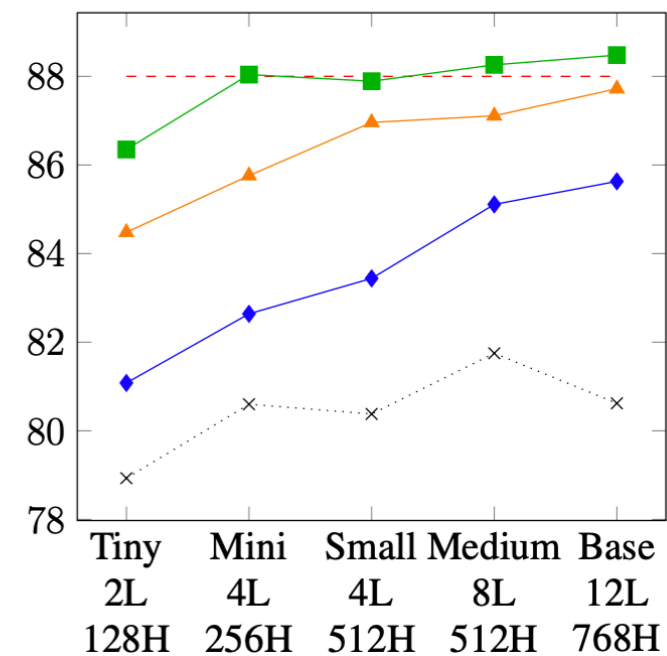
RTE



SST-2

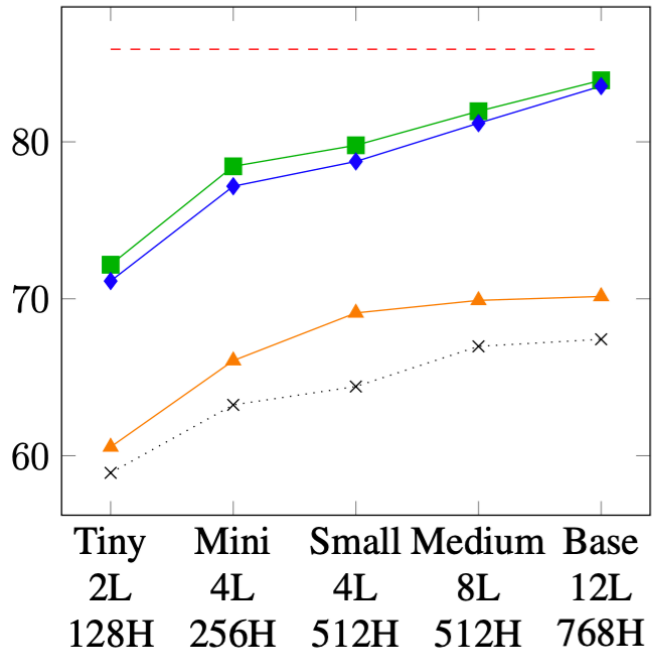


Amazon Book Reviews

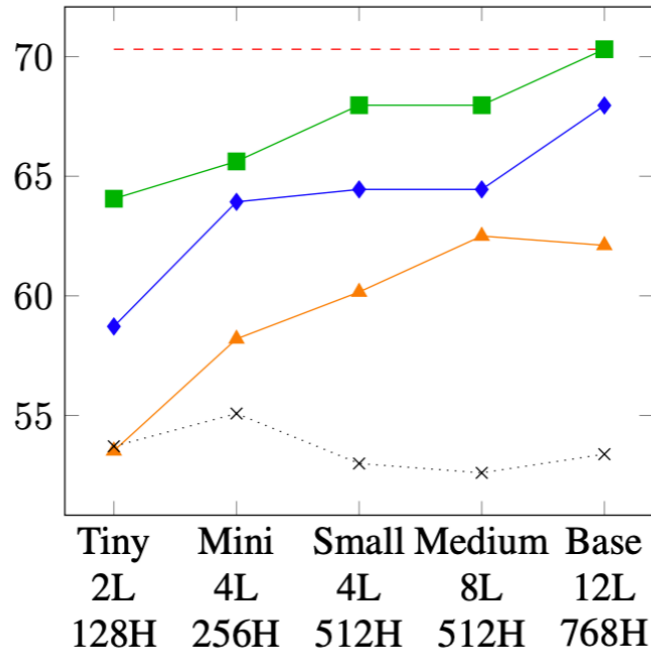


--- Teacher    ■ Pre-trained Distillation    ◆ Pre-training+Fine-tuning    ▲ Distillation    × Basic Training

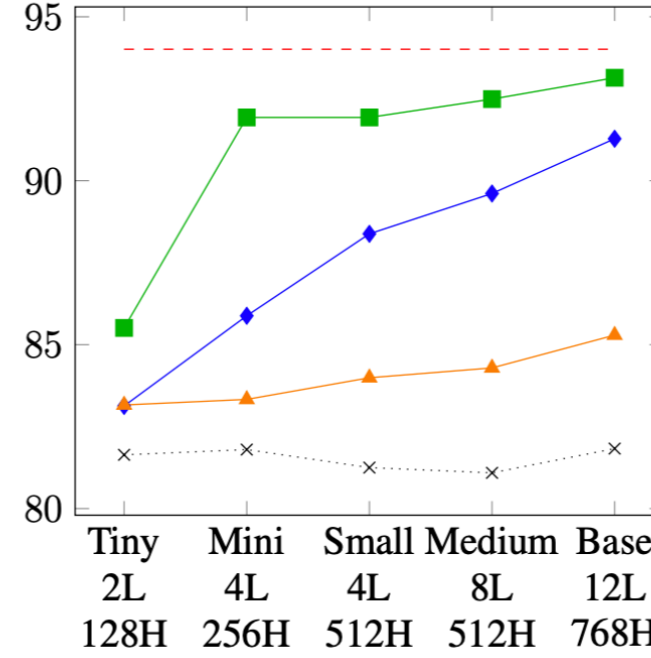
MNLI



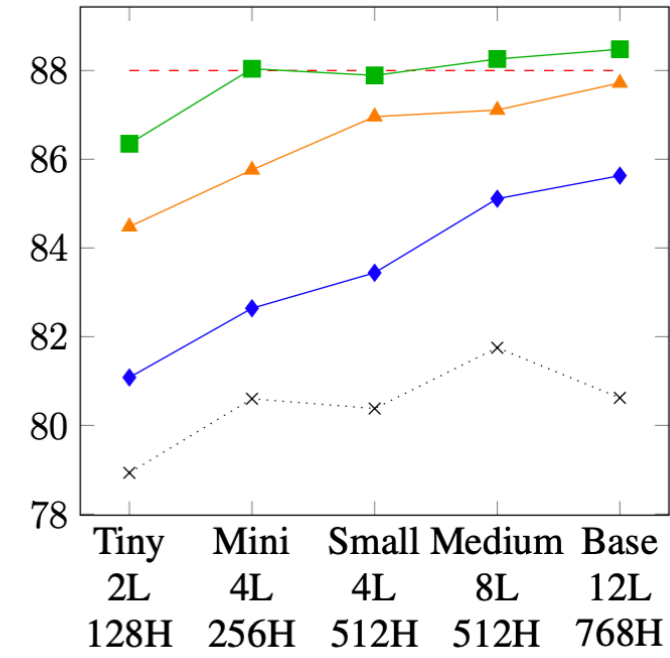
RTE



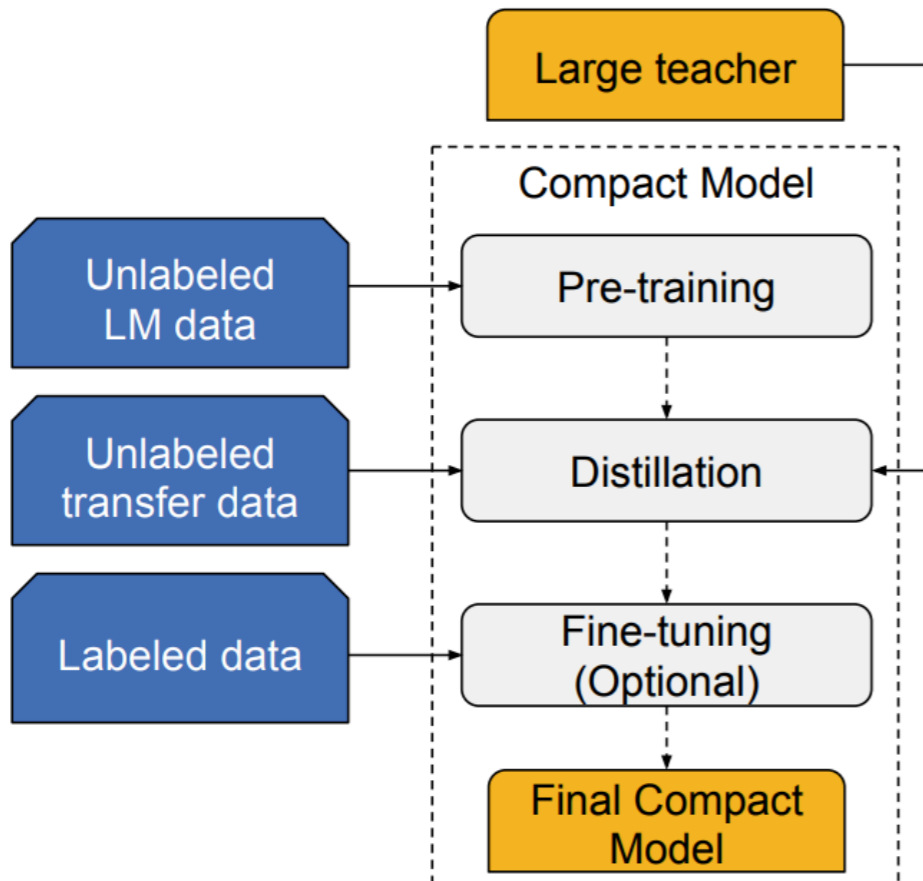
SST-2



Amazon Book Reviews



--- Teacher    ■ Pre-trained Distillation    ◆ Pre-training+Fine-tuning    ▲ Distillation    × Basic Training



**The Lottery Ticket Hypothesis.** *A randomly-initialized, dense neural network contains a subnetwork that is initialized such that—when trained in isolation—it can match the test accuracy of the original network after training for at most the same number of iterations.*

1. Randomly initialize a neural network  $f(x; \theta_0)$  (where  $\theta_0 \sim \mathcal{D}_\theta$ ).
2. Train the network for  $j$  iterations, arriving at parameters  $\theta_j$ .
3. Prune  $p\%$  of the parameters in  $\theta_j$ , creating a mask  $m$ .
4. Reset the remaining parameters to their values in  $\theta_0$ , creating the winning ticket  $f(x; m \odot \theta_0)$ .

How to prune? Simply remove the weights with the lowest magnitudes in each layer

# Can prune a significant fraction of the network with no downstream performance loss

Dataset	MNLI	QQP	STS-B	WNLI	QNLI	MRPC	RTE	SST-2	CoLA	SQuAD	MLM
Sparsity	70%	90%	50%	90%	70%	50%	60%	60%	50%	40%	70%
Full BERT <sub>BASE</sub>	82.4 ± 0.5	90.2 ± 0.5	88.4 ± 0.3	54.9 ± 1.2	89.1 ± 1.0	85.2 ± 0.1	66.2 ± 3.6	92.1 ± 0.1	54.5 ± 0.4	88.1 ± 0.6	63.5 ± 0.1
$f(x, m_{\text{IMP}} \odot \theta_0)$	82.6 ± 0.2	90.0 ± 0.2	88.2 ± 0.2	54.9 ± 1.2	88.9 ± 0.4	84.9 ± 0.4	66.0 ± 2.4	91.9 ± 0.5	53.8 ± 0.9	87.7 ± 0.5	63.2 ± 0.3
$f(x, m_{\text{RP}} \odot \theta_0)$	67.5	76.3	21.0	53.5	61.9	69.6	56.0	83.1	9.6	31.8	32.3

What if you only have access to the model's argmax prediction, and you also don't have access to its training data?

# Thieves on Sesame Street!

## Model Extraction of BERT-based APIs



Kalpesh  
Krishna<sup>1</sup>



Gaurav S.  
Tomar<sup>2</sup>



Ankur P.  
Parikh<sup>2</sup>



Nicolas  
Papernot<sup>2</sup>



Mohit  
Iyer<sup>1</sup>

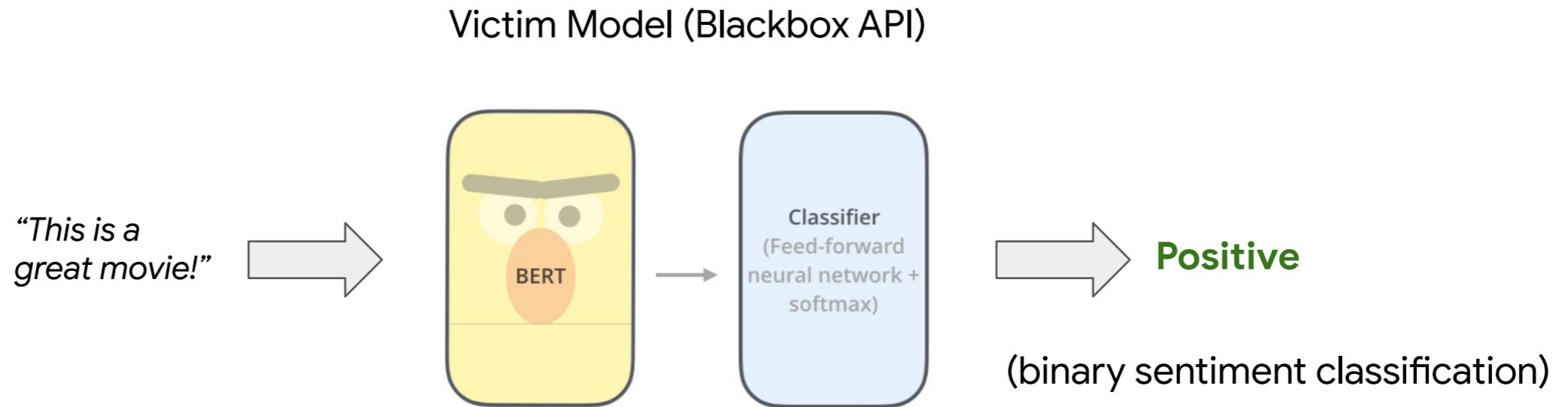
<sup>1</sup> **UMass**  
**Amherst**

<sup>2</sup>  **Google AI**

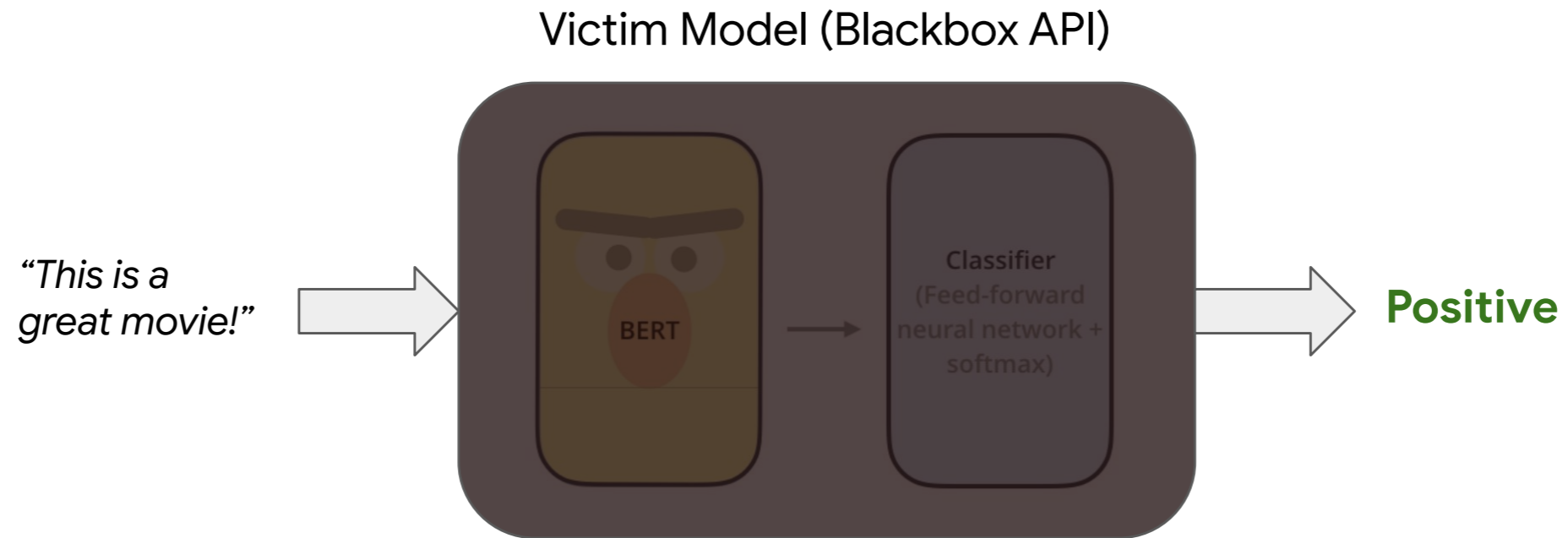
*Work done during an internship at Google AI Language.*



What are model extraction attacks?



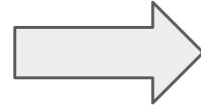
A company trains a binary sentiment classifier based on BERT



It is released as a black-box API (the "victim model")



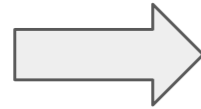
*“seventeen Ill.  
miles Vegas”*



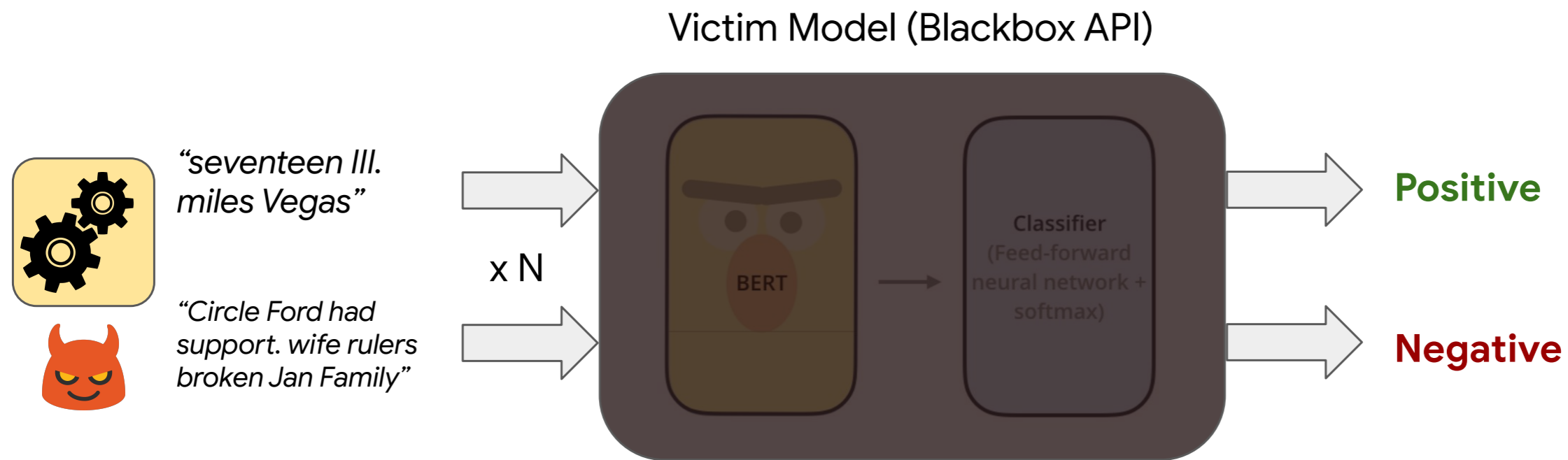
x N



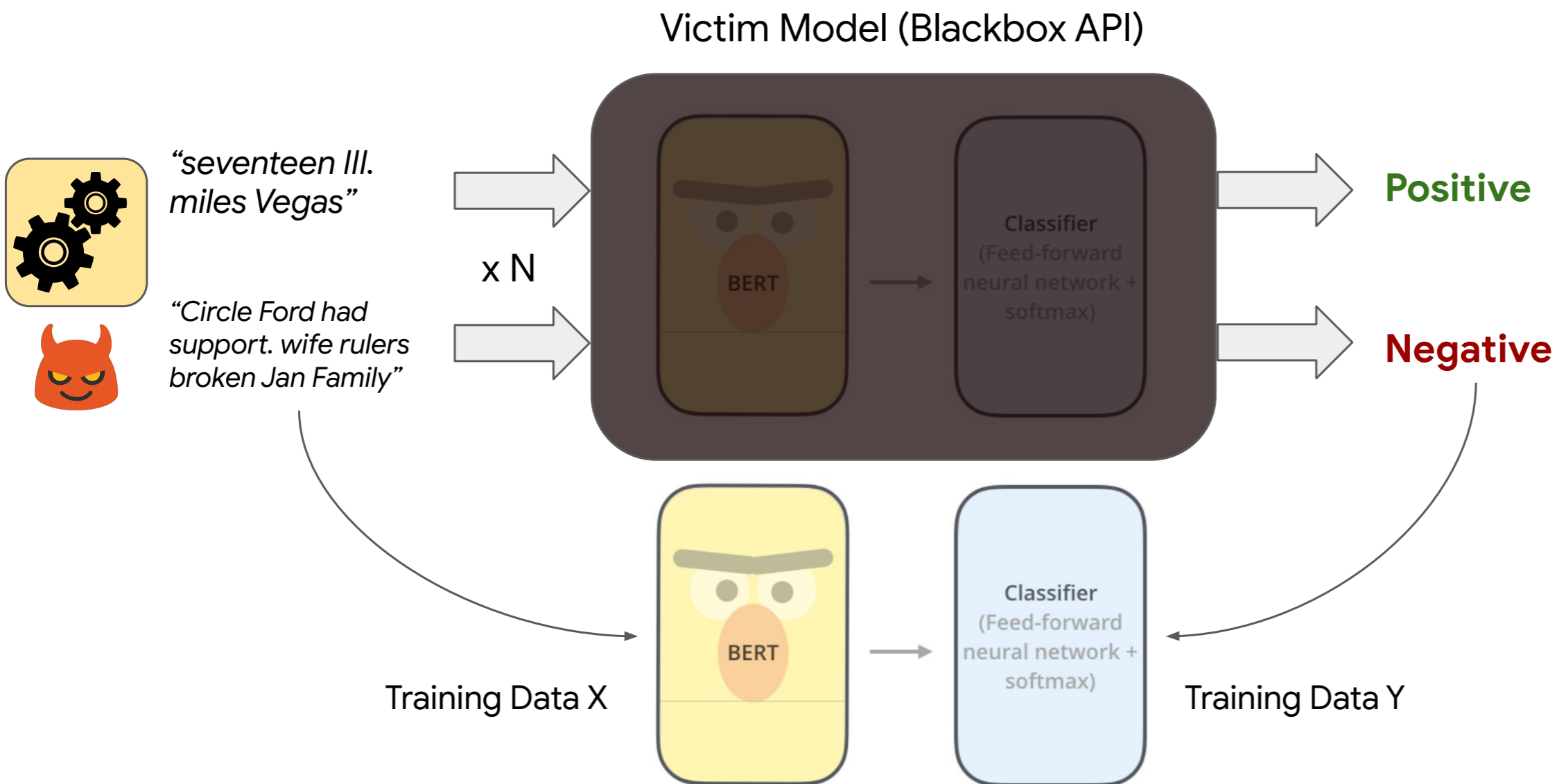
*“Circle Ford had  
support. wife rulers  
broken Jan Family”*



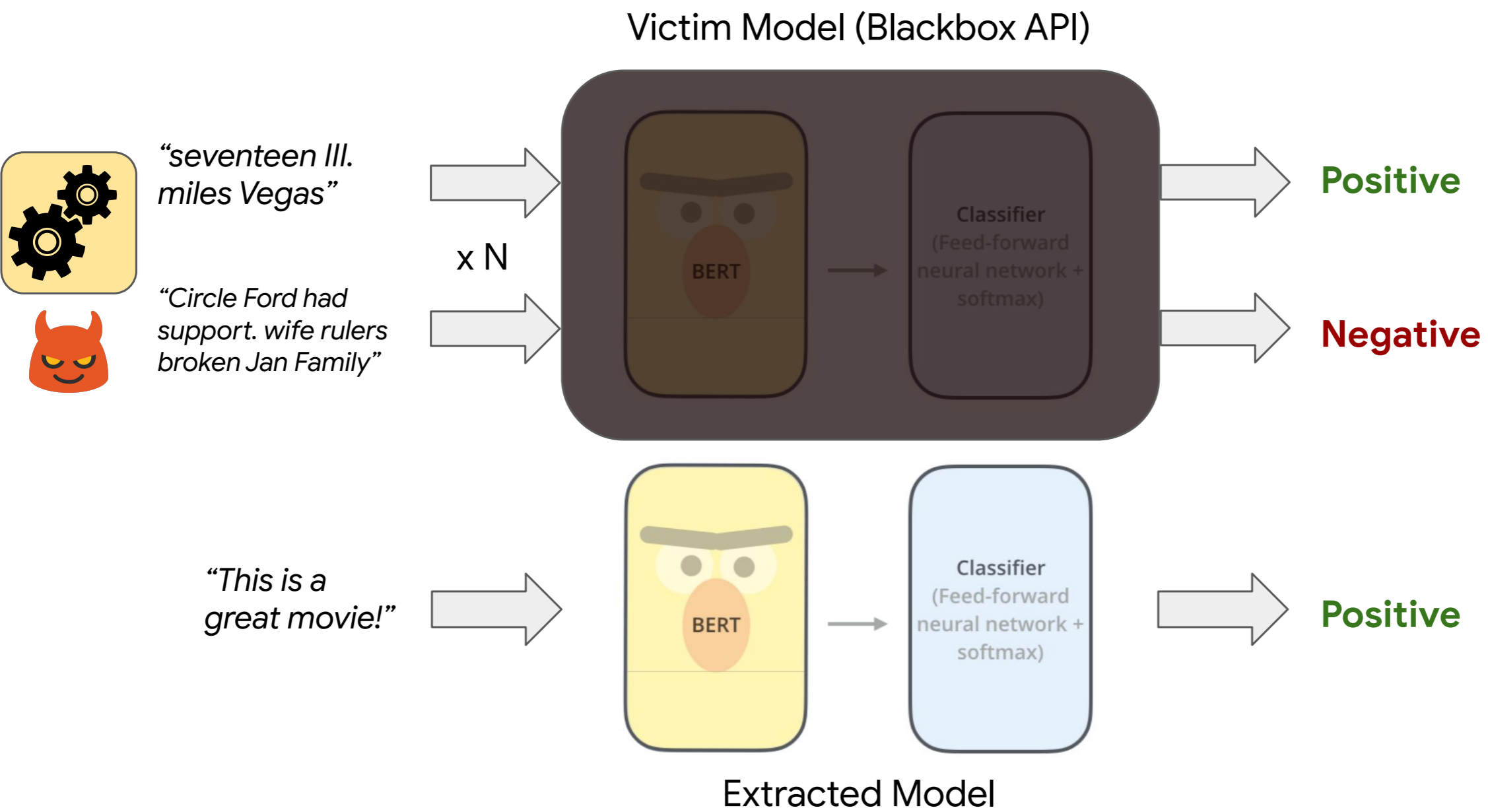
A malicious user generates many queries  
(in this work, **random gibberish sequences of words**)



The attacker queries the API with the generated inputs and collects the labels



The collected data is used to train a “copy” of the model



The stolen copy ("extracted model") works well on real data

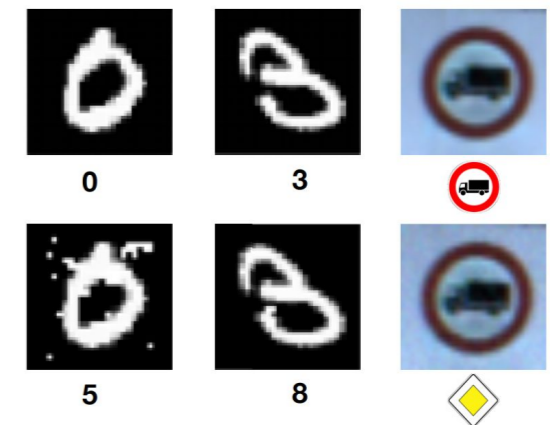
# Why is model extraction a problem?



Theft of intellectual property



Leakage of original training data



Adversarial example generation



# These attacks are economically practical

Google Cloud Natural Language API cost  $\leq$  \$1.00 per 1000 API calls.

<b>Dataset</b>	<b>Size</b>	<b>Upperbound Price</b>
SST2 (sentiment classify)	67349 sentences	\$62.35
Switchboard (speech)	300 hours	\$430.56
Translation	1 million sentences (100 characters each)	\$2000.00

Smart attackers can scrape APIs like Google Translate for free

<https://cloud.google.com/products/calculator/>

# How is this different from distillation?



No training data



Goal is theft, not  
compression

We attack BERT models for,

- 1) sentiment classification (SST2)
- 2) natural language inference (MNLI)
- 3) question answering (SQuAD, BoolQ)

# We use two query generators - RANDOM & WIKI

## RANDOM

(gibberish sequences of words  
sampled from a fixed vocabulary)

1. cent 1977, preparation (120 remote  
Program finance add broader protection
2. Mike zone fights Woods Second State  
known, defined come

## WIKI

(sentences from Wikipedia)

1. The unique glass chapel made public  
and press viewing of the wedding easy.
2. Wrapped in Red was first released  
internationally on October 25, 2013.

For multi-input tasks (like question answering) we ensure inputs are related to each other

**RANDOM Paragraph:** as and conditions Toxostoma storm, The interpreted. Glowworm separation Leading killed Papps wall upcoming Michael Highway that of on other Engine On to Washington Kazim of consisted the " further and into touchdown(AADT), Territory fourth of h; advocacy its Jade woman "lit that spin. Orange the EP season her General of the

For multi-input tasks (like question answering) we ensure inputs are related to each other

**RANDOM Paragraph:** as and conditions Toxostoma storm, The interpreted. Glowworm separation Leading killed Papps wall upcoming Michael Highway that of on other Engine On to Washington Kazim of consisted the " further and into touchdown(AADT), Territory fourth of h; advocacy its Jade woman "lit that spin. Orange the EP season her General of the

**RANDOM Question:** Kazim Kazim further as and Glowworm upcoming interpreted. its spin. Michael as

For multi-input tasks (like question answering) we ensure inputs are related to each other

**RANDOM Paragraph:** as and conditions Toxostoma storm, The interpreted. Glowworm separation Leading killed Papps wall upcoming Michael Highway that of on other Engine On to Washington Kazim of consisted the " further and into touchdown(AADT), Territory fourth of h; advocacy its Jade woman "lit that spin. Orange the EP season her General of the

**RANDOM Question:** What's Kazim Kazim further as and Glowworm upcoming interpreted. its spin. Michael as?

## Results - attacks are effective

	# of Queries	SST2 (%)	MNLI (%)	SQUAD (F1)
<b>API / Victim Model</b>	1x	93.1	85.8	90.6
<b>RANDOM</b>	1x	90.1	76.3	79.1
<b>RANDOM</b>	upto 10x	90.5	78.5	85.8
<b>WIKI</b>	1x	91.4	77.8	86.1
<b>WIKI</b>	upto 10x	91.7	79.3	89.4

A BERT model trained on the real SQuAD data gets 90.6 F1



## Results - attacks are effective

	# of Queries	SST2 (%)	MNLI (%)	SQUAD (F1)
<b>API / Victim Model</b>	1x	93.1	85.8	90.6
<b>RANDOM</b>	1x	90.1	76.3	79.1
<b>RANDOM</b>	upto 10x	90.5	78.5	85.8
<b>WIKI</b>	1x	91.4	77.8	86.1
<b>WIKI</b>	upto 10x	91.7	79.3	89.4

**RANDOM achieves 85.8 F1 (~95% performance) without seeing a single grammatically valid paragraph or question during training**

## Results - attacks are effective

	# of Queries	SST2 (%)	MNLI (%)	SQUAD (F1)
<b>API / Victim Model</b>	1x	93.1	85.8	90.6
<b>RANDOM</b>	1x	90.1	76.3	79.1
<b>RANDOM</b>	upto 10x	90.5	78.5	85.8
<b>WIKI</b>	1x	91.4	77.8	86.1
<b>WIKI</b>	upto 10x	91.7	79.3	89.4

WIKI achieves 89.4 F1 (~99% performance) without seeing a single grammatically valid question during training

## Key findings from experimental analysis

- better pretraining  $\Rightarrow$  better model extraction
- WIKI / RANDOM queries closer to the victim model's learnt distribution are more effective

# What is a good defense?

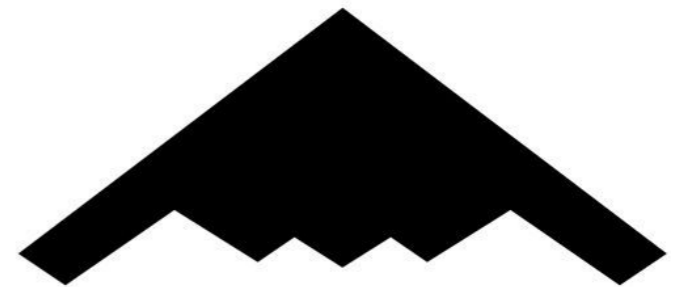


Stops or delays  
model extraction

~~Accuracy = 91.2%~~

Accuracy = 91.1%

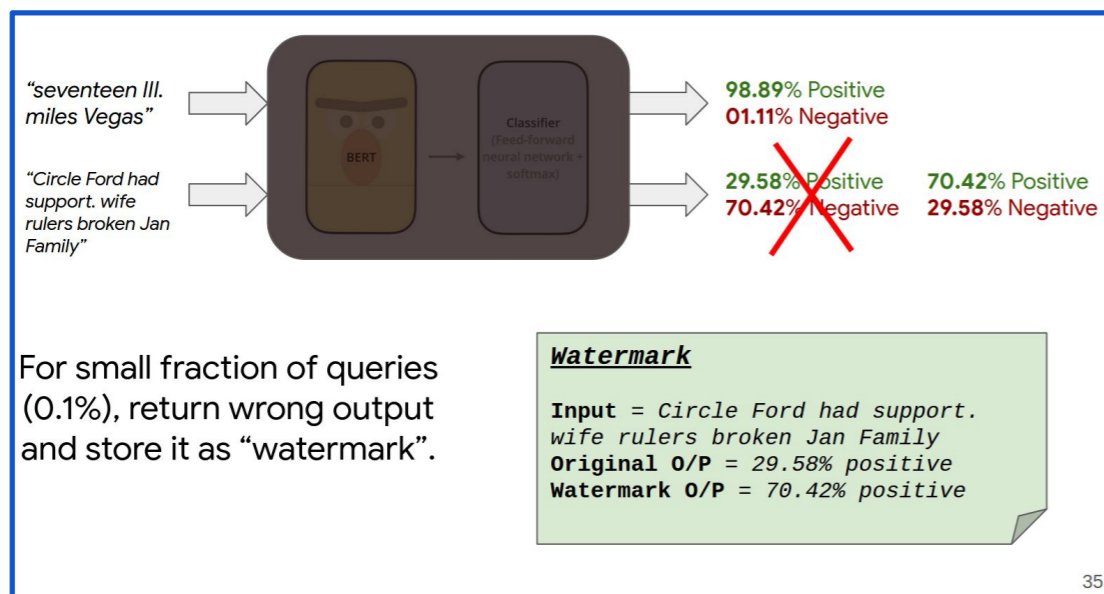
Utility preserving



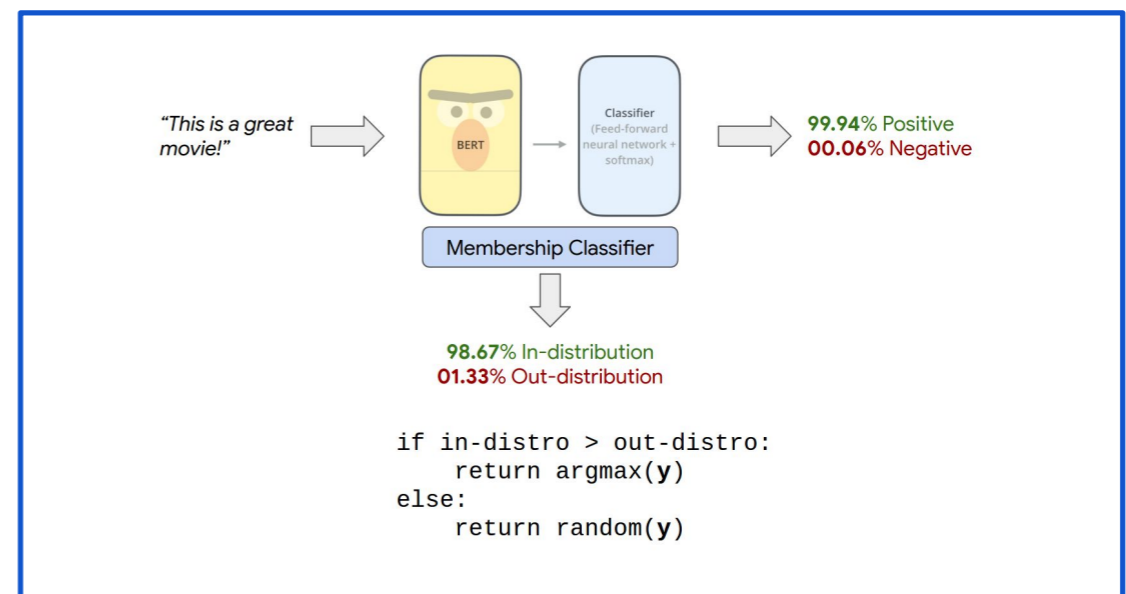
Stealthy, to prevent  
counter-attacks

# What defenses do we investigate in the paper?

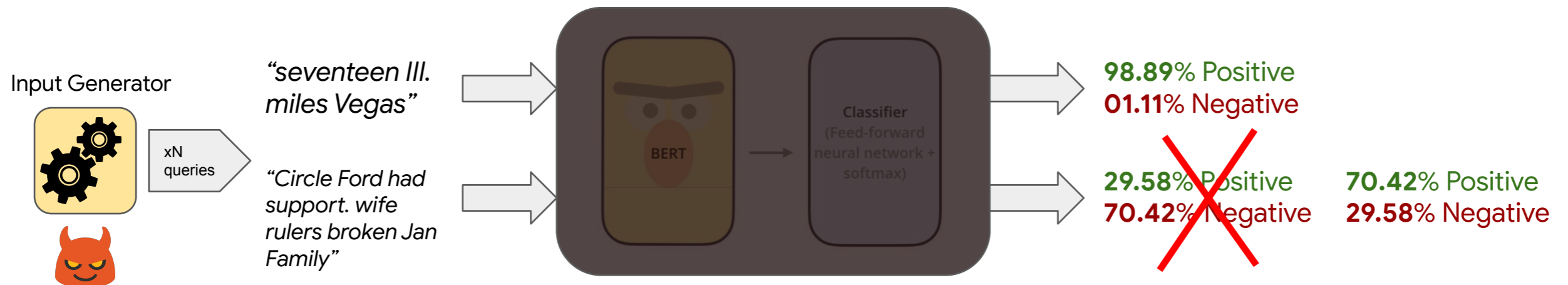
## API Watermarking



## Membership Classification



# What is watermarking?



For small fraction of queries (0.1%), return wrong output and store it as “watermark”.

## Watermark

**Input** = *Circle Ford had support. wife rulers broken Jan Family*  
**Original O/P** = 29.58% positive  
**Watermark O/P** = 70.42% positive

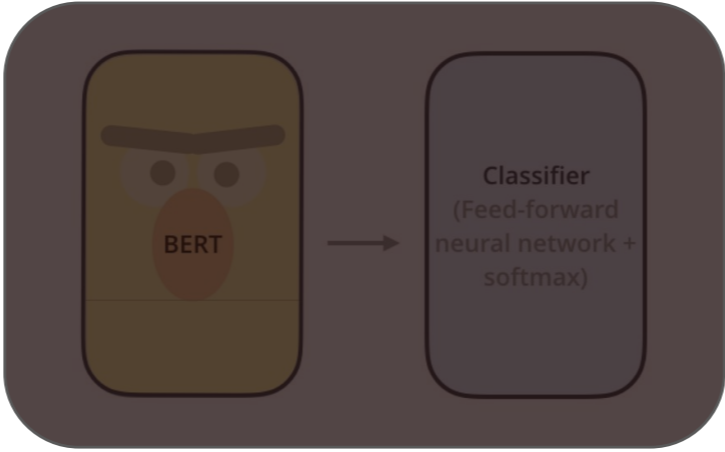
# Watermark verification

Watermark

**Input** = *Circle Ford had support.  
wife rulers broken Jan Family*  
**Original O/P** = 29.58% positive  
**Watermark O/P** = 70.42% positive

Verify watermark if stolen model is released with query access.

“Circle Ford had support. wife rulers broken Jan Family”



65.42% Positive  
34.58% Negative

Stolen Model 

Deep models have high capacity!

Extracted model will memorize watermarks



# Watermarking works well!

Task	Model	Watermark w/ Wrong Labels	Watermark w/ Correct Labels
MNLI	WIKI	2.8%	94.4%
MNLI	watermarked WIKI	52.8%	35.4%
MNLI	watermarked WIKI (10 epochs)	87.2%	7.9%
SQUAD	WIKI	0.0 EM	94.3 EM
SQUAD	watermarked WIKI	5.7 EM	14.9 EM
SQUAD	watermarked WIKI (10 epochs)	74.7 EM	1.1 EM

No significant change in dev accuracy!

# Watermarking works well!

Task	Model	Watermark w/ Wrong Labels	Watermark w/ Correct Labels
MNLI	WIKI	2.8%	94.4%
MNLI	watermarked WIKI	52.8%	35.4%
MNLI	watermarked WIKI (10 epochs)	87.2%	7.9%
SQUAD	WIKI	0.0 EM	94.3 EM
SQUAD	watermarked WIKI	5.7 EM	14.9 EM
SQUAD	watermarked WIKI (10 epochs)	74.7 EM	1.1 EM

No significant change in dev accuracy!

# Watermarking works well!

Task	Model	Watermark w/ Wrong Labels	Watermark w/ Correct Labels
MNLI	WIKI	2.8%	94.4%
MNLI	watermarked WIKI	52.8%	35.4%
MNLI	watermarked WIKI (10 epochs)	87.2%	7.9%
SQUAD	WIKI	0.0 EM	94.3 EM
SQUAD	watermarked WIKI	5.7 EM	14.9 EM
SQUAD	watermarked WIKI (10 epochs)	74.7 EM	1.1 EM

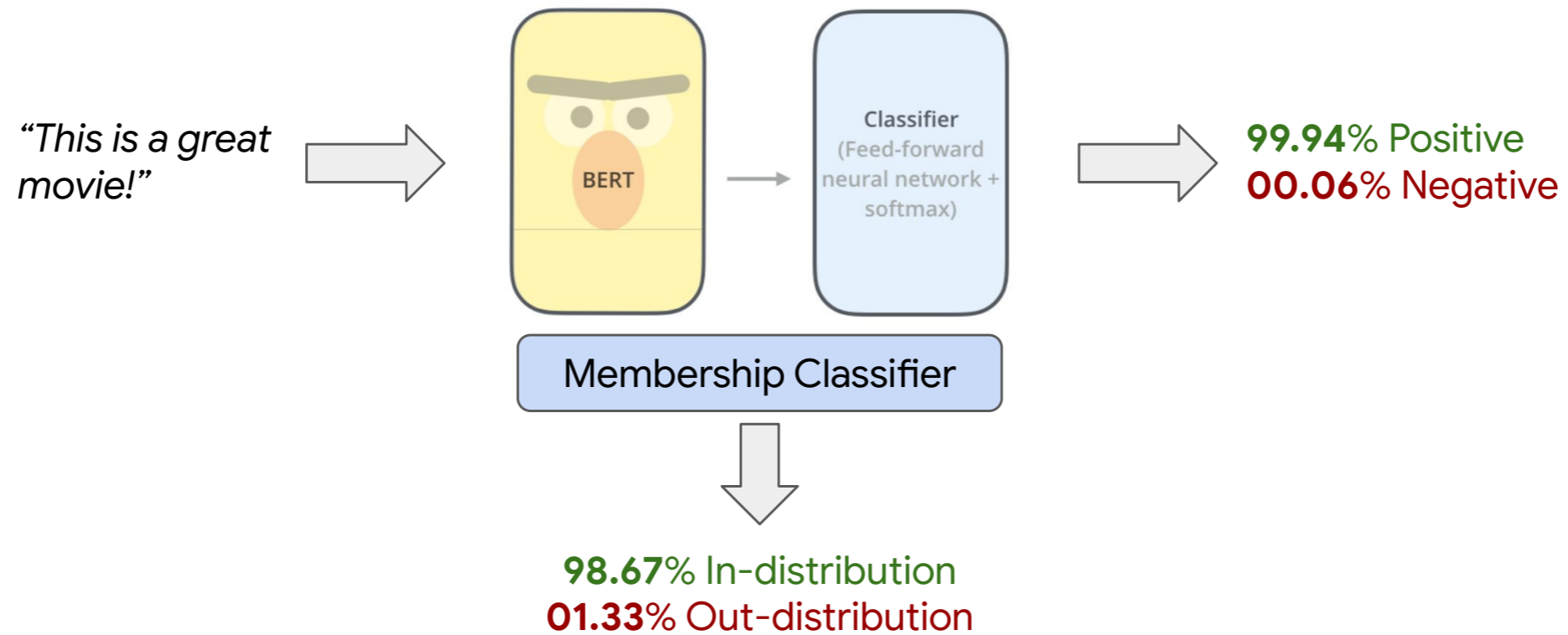
No significant change in dev accuracy!

**Limitation #1**  
watermarking requires public access to  
extracted model

## **Limitation #2**

counter-attack with differentially private training / double extraction possible

# What is membership classification?



```
if in-distro > out-distro:  
    return argmax(y)  
else:  
    return random(y)
```

# Membership Classification

- Train binary classifier with features last layer + logits of trained API
- classify training data vs WIKI attack data
- Evaluate on dev set + auxiliary test sets

<b>Task</b>	<b>Features</b>	<b>WIKI %</b>	<b>RANDOM %</b>	<b>SHUFFLE %</b>
MNLI	last layer + logits	99.34%	99.14%	87.36%
	logits	90.66%	91.20%	82.34%
	last layer	99.15%	99.05%	88.88%
SQuAD	last layer + logits	98.78%	99.97%	99.70%
	logits	81.45%	84.72%	81.99%
	last layer	98.79%	98.89%	98.97%

## **Limitation:**

Genuine queries can be out-of-distribution  
but still sensible

Only works for **RANDOM** queries